HW2

LinxiaoBai

2.3:
The data has total(200+450+300+1500+700+44)=3194 obs, The median will fall in the the 1597th element.
By counting the cumulative sum, the 1597th elements falls in the 21-50 group.

So the median is approximated to be 21-50.

2.6:
Euclidean D:

D^2=(22-2)^2+(1-0)^2+(42-36)^2+(10-8)^2=441
D=21

Manhattan D:

D= (22-2)+(1-0)+(42-36)+(10-8)
=29

Minkowski D:

D^3=(22-2)^3+(1-0)^3+(42-36)^3+(10-8)^3=8225
D=20.186

Chebyshev D:

D=(22-2)=20

2.7:
The most common way of computing median is sorting data then take the middle element.
The time complexity is O(nlog(n)) by mergeSort.

Also, bin smoothing technique can also approximate the mean by first split the data by its value into k bins. This step takes time O(n). Then sort the value, and compute the cumulative sample size, and find the bin that the middle element resides. This step can be as good as O(klog(k)), with k being the number of the bins. The last step is either repeat the process to the bin of which we determine the median resides, or sort the bin and pick the median. So the overall complexity should be max(O(klog(k)), O(n))

Alternatively, dealing with screwed data, and limited options of values, it is often time smarter to group the data by its value and from the cumulative count to determine the compute the median residence. This step is O(n).

Also, if a certain lower bound, or upper bound can be determined that the median will not resides. It is safe to eliminate most of the data and transform the data into finding kth percentile with smaller size of data.

2.8:
This part is computed using python,
a)
Manhattan D:
0.2
0.9
0.4
0.3
0.7

chebyshev D:
0.1
0.6
0.2
0.2
0.6

cosineD:
8.60855604401e-06
0.00424773874711
3.05161812123e-05
0.000971765062438
0.0346366069717

Euclidean D
0.141421356237
0.67082039325
0.282842712475
0.22360679775
0.60827625303

b)
the distance after regularizing and sorting are:
0.0041493508032
0.00781232119311
0.044085486556
0.0921709145784
0.263198050797

3.1:
If the data we are study is an off-line stock market transition data, and the target is to evaluate the mean of the stock price. Then the accuracy should be of major concern, instead of integrity and consistence. Because if the data we collect has a bias due to outside condition. Then the mean we be equally biased. However, even the data is lack of completeness, given a large number of observation, the mean should still be captured.

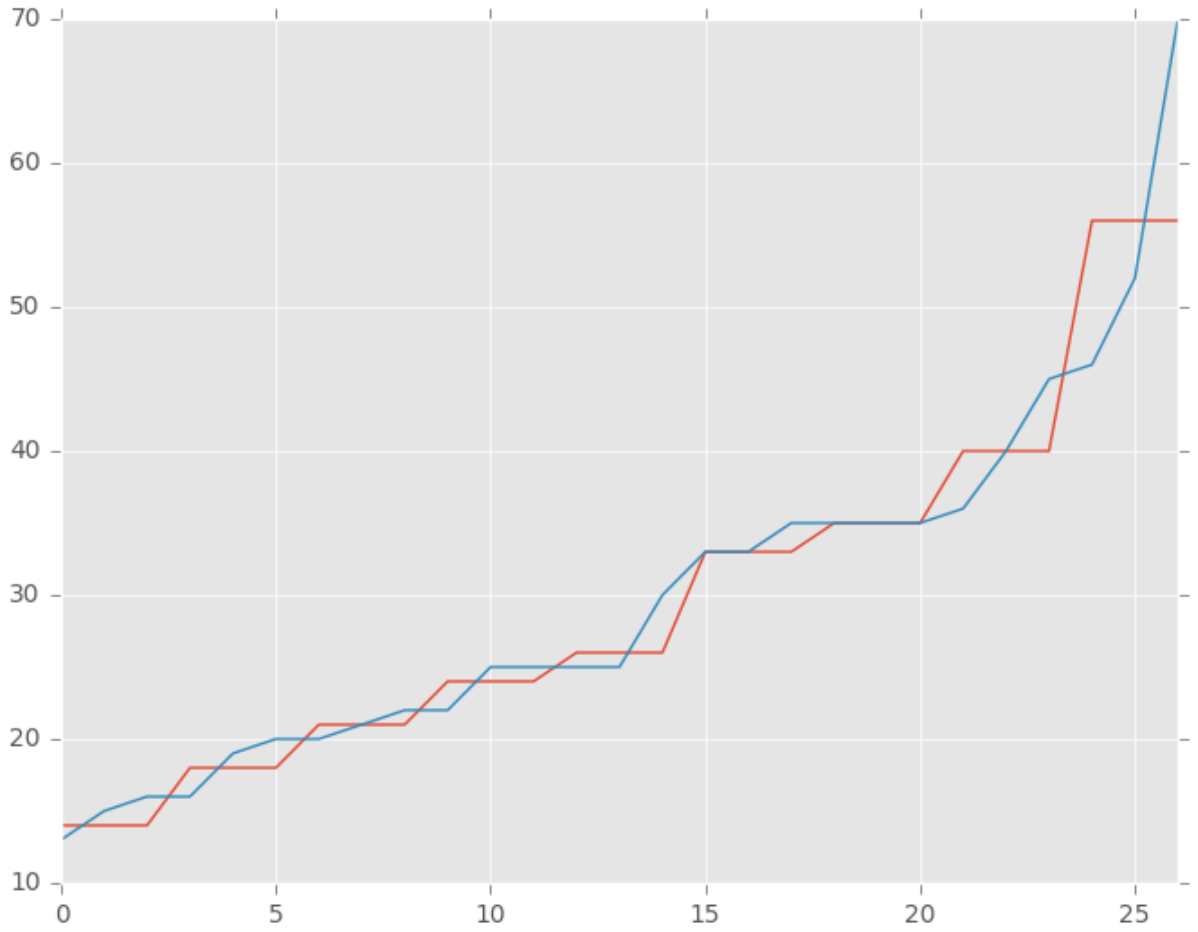
However, if the data is a user cell phone text log, and the goal is to do context-based NLP based on the text message. Then the task should rely highly on the completeness of the data. Because broken sentences and broken dialog will make context unrecognizable.

As for consistency it regards a popularly discussed topic : ID matching, sometimes a single user's behavior cannot be recognized because the id is not matched in different database. Moreover, two different user's behavior can be identified as one person. This type of question is critical when analyze user behavior based on the data from different resources, and database.

The other two quality I provide are:
timeliness
interpretability

3.3

a)



The red line is the line after smoothing, it appears that the line increases is smoother. Especially at the last part, it is clear that the blue line has higher slope than the red line.

b) There are a few of simple techniques that helps determining outliers.
For examples, one can draw boxplot and scatterplot to visually see which of the points are out of the normal range. Also, doing clustering helps seeing the isolated points outside of the cluster. In general, if the distribution where the sample comes from is pre-known, a p-value associated with a hypothesis test can be constructed to determine if the point falls in the normal range. In one-dimension-case if the sample falls in 2-2.5 sd from the sample mean the point can be determined as outlier.

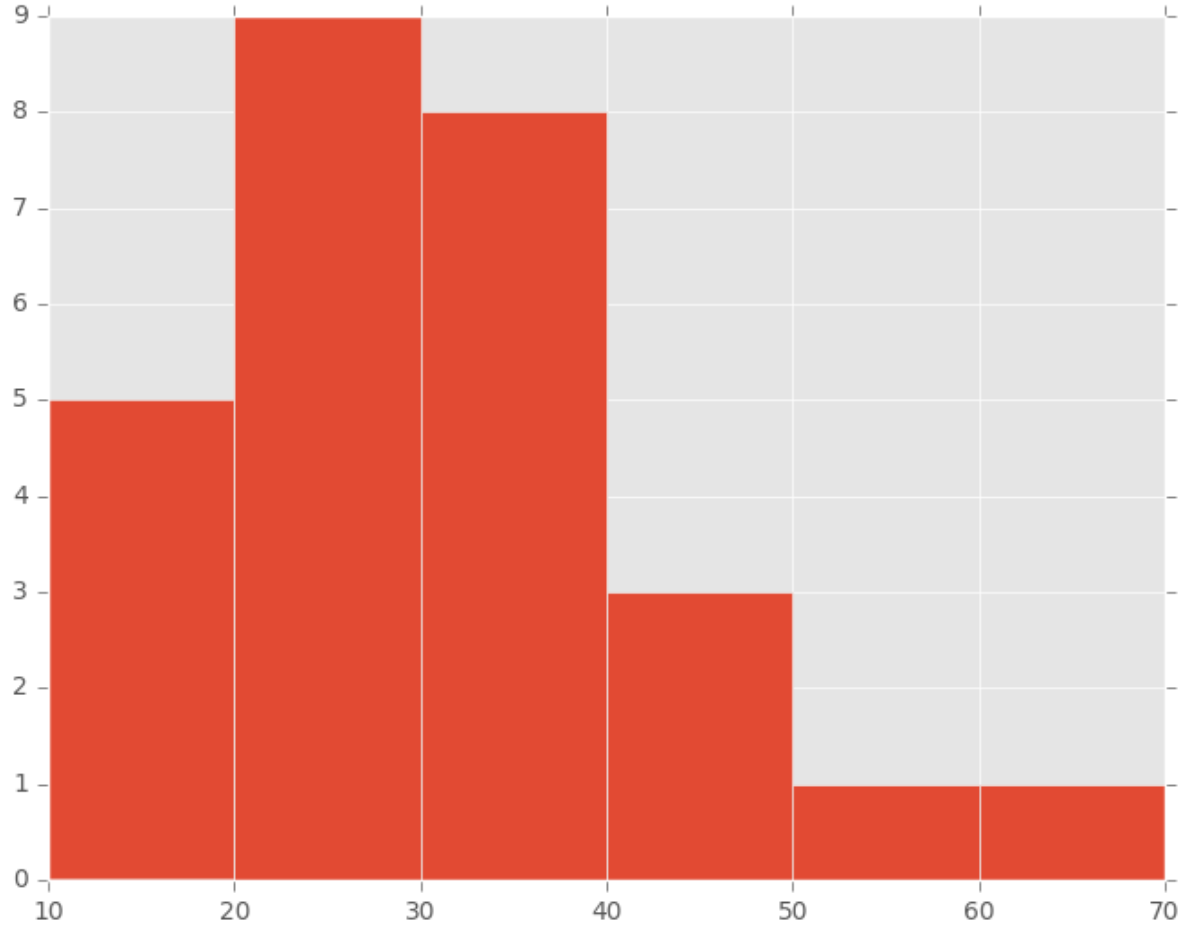
c) regression, binning, and clustering

3.5
a) [new minA,new maxA]
b) negative infinity to infinity
c) [-1,1]

3.7
a)
v=0+(1-0)*(35-13)/(70-13)
=0.38596
b)
given mean=30
sd=13
z=(35-30)/13
=0.384615
c)the greatest number is 70, so divide 35 by 100 and v=0.35

d)personally, the three transformation are pretty much the same to me. After the transformation, the data preserves the same degree of freedom, and all three are of of linear transformation, given certain parameter, the transformation can be reversed back to the original data. However, the first transformation allows the data to be mapped to desired domain. This method is useful in industrial when a normalized data in certain region is desired.

3.11
a)



the plot is shown, with bin-width of 10

b)
SRSWOR example: 15,15,16,18,35

SRSWR example: 20,22,33,35,46
The two SRS above is selected randomly by pick index

Cluster Sample: prune the dataset by each every 5observations. Then randomly select any partition as the result. In the example we select the third partition, with obs of index from 11-15. That is [25,25,25,25,30]

Stratified example. First label the data by young, middle age, and senior based on 30-year-old, 60-year-old criterion. Since the sample size is required to be 5. According to the year-group-size after rounding. Two obs from young and middle are randomly picked and 1 from senior is picked.

So the example is [21,25,33,35,70]

3.13

Using the data from 2.8 to test the result, although the data is sorted but it should not matter because the algorithm treat input as unsorted and return a series of classification code starting 0, with the order of the original series' index.

The test results are shown:

a) [0, 1, 2, 2, 3, 4, 4, 5, 6, 6, 7, 7, 7, 7, 8, 9, 9, 10, 10, 10, 10, 11, 12, 13, 14, 15, 16]
b) [0, 1, 1, 1, 2, 3, 3, 3, 3, 3, 4, 4, 4, 4, 6, 7, 7, 8, 8, 8, 8, 8, 9, 11, 11, 13, 19]
c) [0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4]